CLAIMS

We claim

1. A FIFO buffer providing reduced access times without introducing any latency overhead, comprising:

FIFO means capable of storing 'n' data words, each 'm' bits wide, having an 'm' bit wide data input terminal;

read data selection means connected to data output terminals of the FIFO means and having two data output terminals providing simultaneous access to a selected storage location;

odd read pointer generating means for providing the selection input to the data selection means for selecting data at an odd read address;

even read pointer generating means for providing the selection input to the data selection means for selecting data at an even read address;

multiplexing means coupled to each of the two data output terminals of the read data selection means for selecting one of the outputs of the read data selection means as the final output of the FIFO; and

state controlling means coupled to the multiplexing means for controlling the selection of the final FIFO output and to the odd and even read pointer generating means.

2. The improved FIFO buffer of claim 1, further comprising FIFO status providing means coupled to a selected read pointer means for generating FIFO status signals.

3. The improved FIFO buffer of claim 1, further comprising adder means coupled to the selected read pointer generating means to increment the read address for generating the next read address.

4. The improved FIFO buffer of claim 1 wherein said state maintaining means can have two states namely, odd and even.

5. A method for reducing the access times of a FIFO buffer without introducing latency overhead, comprising the steps of:

providing a FIFO capable of storing 'n' data words, each 'm' bits wide, having an 'm' bit wide data input terminal;

connecting a read data selector to an output of the FIFO and providing simultaneous access to a selected storage location at an odd address and a selected storage location at an even address;

providing selection inputs to the read data selector for selecting an odd read address and an even read address;

multiplexing the output of the read data selector to enable selection of a desired one of the outputs of the read data selector as the final output of the FIFO; and

controlling the state of the FIFO to select one of the multiplexer output as the final output of the FIFO and to control the selection input to the read data selector for selecting an odd read address and an even read address.

6. The method of claim 5, further comprising generating FIFO status signals.

7. The method of claim 5, further comprising generating the next read address by incrementing the current read address.

8. The method of claim 5 wherein said state can be odd or even.

9. A synchronous FIFO buffer, comprising:

a FIFO circuit configured to receive, store, and output data;

a data select circuit coupled to the FIFO circuit to receive data from the FIFO circuit and having a first data output for outputting even data and a second data output for outputting odd data;

a multiplexer circuit coupled to the first and second data outputs of the data select circuit and having a control input and a read data output;

a finite state machine having an output coupled to the control input of the multiplexer circuit, the finite state machine configured to generate a control signal to control the output of the multiplexer circuit; and

a pointer circuit coupled to the finite state machine and configured to generate a read address that is output to the data select circuit.

10. The FIFO circuit of claim 9, wherein the data select circuit is configured to receive a next word from the FIFO circuit and assign it to one of the first data output and the second data output that is not in current use.

11. The FIFO buffer of claim 9, wherein the finite state machine comprises a single D-flip-flop.

12. The FIFO buffer of claim 9, wherein the finite state machine is configured to maintain two states, an odd state and an even state.

13. The FIFO buffer of claim 9, wherein the pointer circuit comprises an odd read pointer circuit and an even read pointer circuit, each coupled to the data select circuit and configured to select an odd read address and an even read address, respectively.

14. A FIFO buffer circuit, comprising: a FIFO circuit configured to receive, store, and output words to a first data bus and a second data bus, and control means coupled to the FIFO circuit and the first and second data outputs and configured to fetch a next word from the FIFO and assign it to one of the first data output and the second data output that is not currently in use.

15. The FIFO buffer of claim 14, wherein the control means comprise a finite state machine coupled to a multiplexer, the multiplexer having the first data output and the second data output as inputs and a read data output as an output.

16. The FIFO buffer of claim 15, wherein the finite state machine comprises a D-flip-flop.

17. A method for reducing access time to a FIFO buffer, comprising:

fetching a next word from a FIFO circuit and assigning it to one of a first data out bus and a second data out bus that is not currently in use.

18. A method for providing access to a FIFO buffer, comprising:

providing simultaneous access to a selected storage location at an odd address and a selected storage location at an even address from an output of a FIFO circuit;

providing selection inputs to a read data selector for selecting an odd read address and an even read address;

multiplexing an output of the read data selector to enable selection of a desired one of the outputs of the read data selector as the final output of the FIFO circuit; and

controlling the state of the FIFO to select one of the multiplexer output as the final output of the FIFO and to control the selection input to the read data selector for selecting an odd read address and an even read address.

19. The method of claim 18, comprising controlling the state of the FIFO with a finite state machine configured to have two states, an odd state and an even state.

20. The method of claim 19, further comprising generating a next read address by incrementing a current read address.